INFORME PRÁCTICA ALGORITMOS VORACES

(PRÁCTICA NÚMERO TRES)

HÉCTOR ALBERCA SÁNCHEZ-QUINTANAR
DARIO ANDRÉS FALLAVOLLITA FIGUEROA

Índice

1. Elementos del problema para adaptar su solución mediante una estrategia voraz.

Tenemos una cantidad de kilos máxima que podemos recoger de una cierta cantidad de pueblos, que varía tras cada ejecución.

Se nos plantea un problema en el que tenemos las siguientes variables:

- Un máximo de **p kilos,** siendo estos los kilos que ofrece cada pueblo.

-Una cantidad de **n poblados,** teniendo en cuenta que el trineo sólo puede recorrer **m poblados** al día.

-Por último, el máximo de kilos que puede cargar el trineo.

Sabido esto, procedemos a usar una estrategia voraz, ya que queremos buscar un camino por los distintos pueblos para recoger una cantidad de kilos. Primero ordenamos los pueblos por sus kilos, de mayor a menor, y, posteriormente recorremos los poblados, añadiendo kilos de comida hasta cargar (o al menos intentarlo) por completo el trineo.

2. Complejidad de los métodos utilizados en la estrategia voraz.

El método utilizado para encontrar un camino mediante el algoritmo voraz es:

```
public static void voraz(ArrayList<Pueblo> posibles, int m, int r) {

    int sumaKilos = 0;

    int aux;
    int i = 0;
    int contM = 0;
    while (sumaKilos < r && i != posibles.size() && contM < m) {

        aux = sumaKilos + posibles.get(i).getKilos();

        if (aux <= r && posibles.get(i).getKilos() != 0) {
            sumaKilos += posibles.get(i).getKilos();
            contM++;
            System.out.println("Pueblo: " + posibles.get(i).getId() + " Kilos: " + posibles.get(i).getKilos());
        }
        aux = 0;
        i++;

    }
    System.out.println();
    System.out.println("Kilos de comida recogidos: " + sumaKilos + ", Poblados Visitados: " + contM);
}
```

La complejidad para este algoritmo mostrado sería O(n), debido a que nuestra estructura principal es un bucle while, siendo la complejidad de este O(n). Las complejidades de las demás partes de este método son despreciables.

Para reducir la memoria utilizada, imprimimos directamente los resultados en lugar de almacenarlos para mostrarlos después.

Sobre este método no hay muchos más que añadir.

3. Demostración de que nuestra estrategia NO es óptima.

El algoritmo que nosotros hemos planteado para solucionar este problema NO es óptimo según el siguiente ejemplo.

Con los siguientes datos:

- ➔ Tamaño máximo que puede soportar el trineo: 74 kilos
- ➔ Número de pueblos posibles a visitar: 6
- ➔ Conjunto de kilos de cada pueblo: 20 7 20 13 20 7

Mediante el algoritmo planteado, la solución sería: 20 20 20 13 (debido a su ordenación de mayor a menor aplicada previamente), siendo el peso en el trineo 73. Una solución óptima sería 20 20 20 7 7, ocupando así el tamaño del trineo completamente. Debido a este contraejemplo expuesto, no podemos decir que hayamos construido un algoritmo óptimo.